

FAST EFFICIENT ALGORITHM FOR ENHANCEMENT OF LOW LIGHTING VIDEO

Xuan Dong, Guan Wang*, Yi (Amy) Pang, Weixin Li*, Jiangtao (Gene) Wen, Wei Meng, Yao Lu**

Department of Computer Science, Tsinghua University, Beijing, China

dongxuan8811@gmail.com,

pangy@mails.tsinghua.edu.cn,

jtwen@tsinghua.edu.cn

mengw08@mails.tsinghua.edu.cn

*Department of Computer Science and Technology, Beihang University, Beijing, China

{wg1989john, lexiwzx}@gmail.com

**Department of Electronic Engineering, Tsinghua University, Beijing, China

luyao11175@gmail.com

ABSTRACT

We describe a novel and effective video enhancement algorithm for low lighting video. The algorithm works by first inverting an input low-lighting video and then applying an optimized image de-haze algorithm on the inverted video. To facilitate faster computation, temporal correlations between subsequent frames are utilized to expedite the calculation of key algorithm parameters. Simulation results show excellent enhancement results and 4x speed up as compared with the frame-wise enhancement algorithms.

Index Terms— low lighting video enhancement, de-hazing

1. INTRODUCTION

As video cameras become increasingly widely deployed, the problem of video enhancement for low lighting video has also become increasingly acute. This is because although camera and video surveillance systems are expected to work in all lighting and weather conditions, the majority of these cameras are not designed for low-lighting, and therefore the resulted poor capture quality often renders the video unusable for critical applications. Although infrared cameras are capable of enhancing low-light visibility in [1] and [2], they suffer from the common disadvantage that visible objects must have a temperature higher than its surroundings. In many cases where the critical object has a temperature similar to its surroundings, e.g. a big hole in the road, infrared cameras are not very useful. On the other hand, conventional image processing and radio processing techniques such as histogram equalization may not work well either in many cases, especially for real time processing of video sequences.

In this paper, we propose a novel, simple and effective enhancement algorithm for low lighting video. We show that after inverting the low-lighting video, the pixels in the sky and distant background regions of the inverted video always have high intensities in all color (RGB) channels but

those of non-sky regions have low intensities in at least one color channel, similar to the case of video captured in hazy weather conditions. Therefore, we apply state-of-the-art image de-hazing algorithms to the inverted video frames and show that the combination of inverting low-lighting videos and applying de-hazing on them is an effective algorithm for enhancing low-lighting videos. To facilitate faster implementation, we further utilize temporal correlations between subsequent video frames for the estimation of key algorithm parameters, resulting in 4x speed-ups as compared with the conventional frame-by-frame approach.

2. ENHANCEMENT OF LOW LIGHTING VISION VIDEO

We notice that low lighting video enhancement has much in common with video haze removal, because after inverting the input low-lighting videos, the resulted videos look very much like videos acquired in hazy lighting conditions. Figures 1 and 2 provided some examples of low lighting videos, inverted low lighting videos and videos acquired in hazy lighting conditions.

As mentioned in [3], in hazy conditions, the intensities of background pixels are always high in all color channels, but the intensities of the main objects, e.g. houses, vehicles and people are usually low in at least one channel because of the color, shadow and etc.. To test whether the inverted low lighting video has the same property, we compute the minimum intensity of all color (RGB) channels for each pixel of the inverted videos/images for 30 randomly selected (by Google) low lighting ones, some of which are shown in the top row of Figure 2, as well as 30 random haze videos/images as shown in the bottom row of Figure 2. Figure 3(a) shows the histogram of the minimum intensities of all color channels of all pixels for the 30 inverted low lighting videos/images and Figure 3(b) is the same histogram of the 30 haze videos/images. Comparing Figures 3(a) and 3(b), we find that the two histograms exhibit great similarities, and that more than 80% of pixels in both the inverted and the haze cases have high intensities in all color

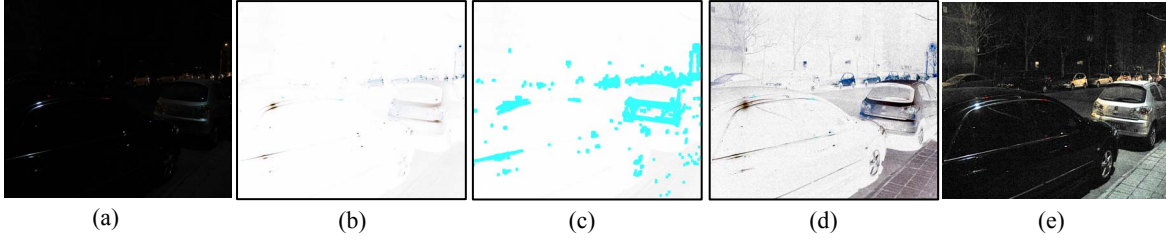


Fig. 1. (a) low lighting video input I . (b) inverted result R from input I . (c) marked image: pixels with low intensity in at least one color (RGB) channel are in green. (d) de-haze output J using equation (7). (e) final output E after inverting image J .

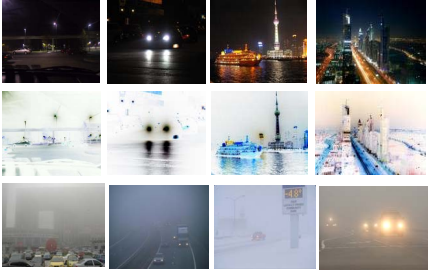


Fig. 2. Top: low lighting examples. Middle: inverted outputs of the Top ones. Bottom: haze examples.

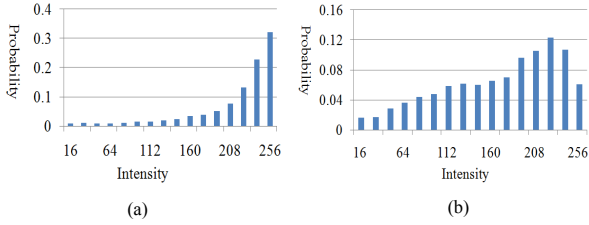


Fig. 3. histogram of all pixels' color channels' minimum intensities of (a) the 30 inverted low lighting videos/images and (b) the 30 haze videos/images.

channels (termed “high intensity pixels”), including almost all of the pixels in sky regions. As for the pixels of non-sky regions, the intensities are always lower in at least one of the color channels (termed “normal intensity pixels”). In Figure 1(c), we mark the normal intensity pixels in green. A pixel is designated as a normal intensity pixel if one of its three color channels' intensities is lower than 180. Clearly, most of the marked pixels are in a non-sky region or object, e.g. vehicles, sidewalks, lights and etc.

The above statistics strongly support our intuition that the inverted low lighting videos are similar to haze videos. Based on the above observation, we propose a novel low lighting video enhancement algorithm by applying the invert operation on low lighting video frames, and then performing haze removal on the inverted video frames, before performing the invert operation again to

obtain the output video frames. Thus, for the input low lighting video frame I , we invert it using:

$$R^c(x) = 255 - I^c(x), \quad (1)$$

where c is the color channel (RGB). $I^c(x)$ is the intensity of a color channel of pixel x of the low lighting video input I .

$R^c(x)$ is the same intensity of inverted image R . The haze removal algorithm we use is based on [3], in which a hazy

image is modeled as shown in (2):

$$R(x) = J(x)t(x) + A(1-t(x)), \quad (2)$$

where A is the global atmospheric light. $R(x)$ is the intensity of pixel x the camera catches. $J(x)$ is the intensity of the original objects or scene. $t(x)$ describes how much percent of the light emitted from the objects or scene reaches the camera. The model is also mentioned in [4-7]. The critical part of all haze removal algorithms is to estimate A and $t(x)$ from the recoded image intensity $I(x)$ so that they can recover the $J(x)$ from $I(x)$. When the atmosphere is homogenous, the $t(x)$ can be expressed as:

$$t(x) = e^{-\beta d(x)}, \quad (3)$$

where β is the scattering coefficient of the atmosphere, and $d(x)$ is pixel x 's scene depth. In the same image, where the β is constant, $t(x)$ is determined by $d(x)$, the distance between the object and the camera. In [3] $t(x)$ is estimated using:

$$t(x) = 1 - \omega \min_{c \in \{r, g, b\}} \left(\min_{y \in \Omega(x)} \left(\frac{R^c(y)}{A^c} \right) \right), \quad (4)$$

where ω is 0.8 in this paper. $\Omega(x)$ is a local block centered at x and the block size is 9 in this paper. We utilize this algorithm to estimate $t(x)$ in this paper. To estimate the global atmosphere light A , we first select 100 pixels whose minimum intensities in all color (RGB) channels are the highest in the image, and then among the pixels, we choose the single pixel whose sum of RGB values is the highest. The RGB values of this pixel RGB values are used for A . Thus, according to the equation (2), we can of course recover the $J(x)$ as follows:

$$J(x) = \frac{R(x) - A}{t(x)} + A, \quad (5)$$

however, we find that direct application of equation (5) might lead to under-enhancement for low-lighting areas. To further optimize the calculation of $t(x)$, we focus on enhancing the Region of Interests such as houses, vehicles and etc. while avoid processing background e.g. sky regions in low lighting videos. To adjust $t(x)$ adaptively while maintaining its spatial continuity, so that the resulted video becomes smoother visually, we introduce a multiplier $P(x)$ into (5), and through extensive experiments, we find that $P(x)$ can be set as:

$$P(x) = \begin{cases} 2t(x), & 0 < t(x) < 0.5 \\ 1, & 0.5 < t(x) < 1 \end{cases}, \quad (6)$$

then the recovery equation becomes:

$$J(x) = \frac{R(x) - A}{P(x)t(x)} + A, \quad (7)$$

the idea behind equation (7) is the following. When $t(x)$ is smaller than 0.5, which means that the corresponding pixel needs boosting, we assign $P(x)$ a small value to make $P(x)t(x)$ even smaller so as to increase the RGB intensities of this pixel. On the other hand, when $t(x)$ is greater than 0.5, we refrain from overly boosting the corresponding pixel intensity.

For low-lighting video, once $J(x)$ is obtained, the invert operation of (1) is performed again to produce the enhanced image E of the original input low light video frame. This process is conceptually shown in Figure 1.

3. ACCELERATION OF PROPOSED VIDEO ENHANCEMENT PROCESSING ALGORITHM

3.1. Acceleration Using Motion Estimation (ME)

In this section we propose an approach to accelerate the processing of the algorithm in the previous section. In experiment, we find calculating $t(x)$ occupies nearly 70% of the computation, which means the focus of acceleration technique becomes accelerating the calculation of $t(x)$. Our technique is based on the correlations between temporally successive frames. Since $t(x)$ is decided by the intensities of pixels of each frame, the values of $t(x)$ of co-located pixels should also be temporally correlated in temporally neighboring frames. In particular, if the values of $t(x)$ are stored from the previous frame, and if the pixels in the current frame are determined to be sufficiently similar to the co-located pixels in the previous frame, the stored values

could be used as the $t(x)$ for the corresponding pixels in the current frame, thereby by-passing the calculation of a significant portion of the $t(x)$ values.

To determine if a pixel is sufficiently similar to its co-located counter part in the previous frame, we use the same technique in the motion search process of video coding. We first divide the input frames into Groups of Pictures (GOPs), with each GOP starting using an Intra coded frame (I frame), in which all $t(x)$ values are calculated. In the remaining frames, i.e. P frames, ME is performed. In detail, each frame is divided into non-overlapping 16x16 macro blocks (MBs), for which a motion search using the distortion criterion Sum of Absolute Differences (SAD) are conducted. If the SAD is below the predefined threshold, the MB will be encoded in inter-mode and the calculation of $t(x)$ for the entire MB is skipped by assigning the “best” match MB’s $t(x)$ directly. Otherwise, the MB will be encoded in intra-mode and $t(x)$ still needs to be calculated. In both cases, the values for the current frame are stored for possible use in the next frame.

We take two ME algorithms in this paper. First, we propose a simple ME method named Co ME which only calculates the distortion value of the (0, 0) motion vector for the current MB. If the SAD value is larger than a predefined threshold T , the $t(x)$ of the current MB is calculated. Otherwise, the calculation is by-passed. The Co Me algorithm is easy to implement and cost little storage space. The other ME method is EPZS ME algorithm proposed in [12], which is more difficult to implement and occupies more storage. However, it performs better than Co ME in total speedup because it can provide more accurate ME information to match the reference frames thus to reduce more calculation of $t(x)$. In EPZS ME algorithm, the first-stage SAD threshold T is fixed and predefined. The following threshold changes as follow:

$$T_k = a_k \times \min(MSAD_1, MSAD_2, \dots, MSAD_n) + b_k, \quad (8)$$

where a_k and b_k are fixed value.

Based on the above algorithm, we define the speed up ratio as:

$$Speedup = \frac{t_o}{t_A} = \frac{t_{other} + g \cdot t_i}{t_{other} + t_i + \sum_{k=1}^{g-1} t_p(k)}, \quad (9)$$

t_{other} is the time of enhancement processing except calculating $t(x)$. g is the GOP size. $t_p(k)$ is the time of calculating $t(x)$ for the k^{th} frame in a GOP. t_i is the time of calculating $t(x)$ for the I frame in a GOP. $t_p(k)$ varies because of different content of P frames.

To analyze the equation of speedup more easily, we make two assumptions. Using the average time \bar{t}_p of calculating $t(x)$ for the k^{th} frame in a GOP substitutes for $t_p(k)$. Comparing the time of calculating $t(x)$, we ignore t_{other} . (8) then becomes:

$$Speedup = \frac{g \cdot t_i}{t_i + g \cdot t_p} = c - \frac{c^2}{g + c}, \quad (10)$$

Where $c = \frac{t_i}{t_p}$.

For the same video sequence where t_i is const and t_p is affected by the SAD threshold, frame content and GOP size, speedup is an inverse proportion function of g . We run the proposed acceleration algorithm for the same video sequence with different GOP sizes, and the result is shown in Figure 4. The inverse proportion function fitting curve matches the experimental results very well. Moreover, equation (9) explains how the speedup changes with different GOP sizes and conducts us to set the appropriate GOP size for a needed speedup and quality of the enhanced video.

In addition, our experiment shows after adopting the above algorithm the computational time of $t(x)$ dwindles a lot while the ME time occupies much of the whole processing time. Thus, in next subsection, we propose a fast SAD algorithm to accelerate ME.

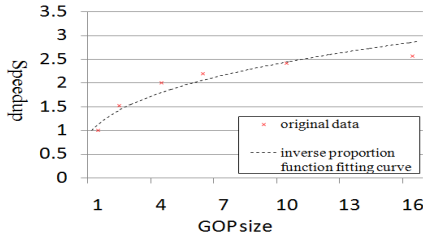


Fig. 4. Speedup of proposed fast enhancement algorithm.

3.2. Acceleration Using Fast SAD Algorithm

Experimental results demonstrate the calculation of SAD takes 60 percent of the whole ME time, i.e. 40 percent of computation cost of the whole low lighting enhancement algorithm. Thus, it is desired for us to propose a fast SAD algorithm.

Some existing fast SAD algorithms are based on advanced hardware e.g. GPU. However, as discussed in [8, 9, 10], in BBME, all the distortion criterions including SAD are based on the inherent assumption that motion fields of video sequences are usually smooth and slowly varying and

each MB contains only one rigid body and traditional motion. In most cases where the assumption is correct, the conventional distortion criterion seems heavy and unnecessary. Moreover, in some other cases, it will not be the true motion vector when the SAD gets its minimum if there is more than one rigid body or the motion is untraditional. In these cases, the more pixels are sampled the greater error the SAD algorithm will probably lead to. However, if we adopt sub-sampling method in our fast SAD algorithm, not only the unnecessary computation in most cases can be reduced but also the poor performance of BBME in complex motion conditions can be improved. Although the fewer samples will reduce the horizontal and vertical solution and have an effect on the accuracy, the error turns out to be slight and acceptable in our algorithm if users do not have strict requirement of video quality but need the algorithm as fast as possible.

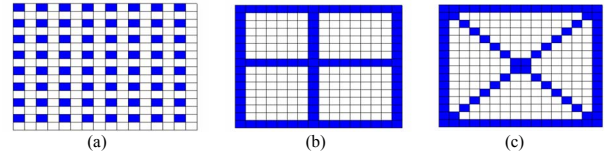


Fig. 5. Fixed subsampling patterns of fast SAD algorithm (a) standard 4:1 pattern. (b) Three-Coefficient pattern. (c) proposed pattern in this paper

There are many subsampling patterns proposed e.g. the adaptive pixel decimation in [9], the standard 4:1 pattern in [10], and etc.. In experiments, we find fixed pixel subsampling pattern is simple and can gain the best speedup in our algorithm. Some existing fixed patterns are the standard 4:1 pattern in [10] shown in Figure 5(a), and the Three-Coefficient pattern in [11] shown in Figure 5(b). We notice in those inferior enhanced MBs, the drawback pixels are always the edge pixels thus an unbalanced subsampling pattern focusing more on edge pixels such as the one shown in Figure 5(b) is more desirable. To make the subsampling pattern symmetrical, we propose an innovative pattern shown in Figure 5(c). The sample we take are the diagonal pixels and the edge pixels. The total number is 60 pixels for 16×16 MB. Experiment results show our algorithm performs as fast as the 4:1 pattern while the effects are not heavily affected. Moreover, in comparison with the exhaustive pixel-wise SAD method, the proposed fast SAD method even gains better enhancement quality with only 20 percent of the computation.

4. EXPERIMENTAL RESULTS

In our experiments, we use Windows PC with the Intel Core 2 Duo processor (2.0GHz) with 3G of RAM. The videos'

resolution is 640×480 .

Examples of the enhancement results of the proposed algorithm in Section 2 are shown in Figures 6, 7 and 8. The original low lighting video inputs i.e. Figure 6(a), 7(a) and 8(a) exhibit very little scene information due to the low illumination level. The enhanced results are shown in Figure 6(b), 7(b) and 8(b) and the red arrows point to areas with notable enhancement. In Figure 6(b), the improvement in visibility is obvious, including four vehicles in the background and the license plate of the vehicle in the foreground. More enhancement results are shown in Figure 7 and 8.

In addition, we perform the video enhancement algorithm using both the Co ME and the EPZS ME with different GOP size, SAD threshold parameters, a_k and b_k . In comparison with the result of the frame-wise enhancement algorithm in which all $t(x)$ values of each frame are calculated, the speedup results are shown in Figure 10(a), and the corresponding PSNR results are shown in Figure 11(a) (GOP size is 10) and 11(b) (GOP size is 30). From Figure 11 we can find the PSNR of the EPZS ME with $T=1000$, $a_k=0.8$ and $b_k=40$ is even higher than the PSNR of Co ME with $T=500$ in the same GOP size. Meanwhile, as shown in Figure 10(a), the speedup of EPZS ME is much higher than that of Co ME too. This demonstrates that the EPZS ME helps find more accurate reference MBs, resulting in much more of the calculation of $t(x)$ bypassed, although the EPZS ME takes more computation costs than the Co ME.

On the other hand, we perform the video enhancement algorithm using the Co and EPZS ME with the fast SAD algorithm. The speedup results are shown in Figure 10(b). The corresponding PSNR results are shown in Figure 12(a) (GOP size is 10) and 12(b) (GOP size is 30). From Figure 6(b), we can learn the fast SAD algorithm enhances the speedup further for both ME algorithms. Meanwhile, the Figure 8 shows the PSNR of EPZS ME using fast SAD algorithm is even enhanced sometimes. To sum up, the fast SAD algorithm benefits the EPZS ME in both speedup and video quality thus results in our low lighting video enhancement algorithm faster and higher quality.

Examples of the results of the acceleration algorithm are shown in Figure 9. Figure 9(a) is the 16th frame of the input video sequence, Figure 9(b) and 9(c) are the enhanced results of the same frame using our acceleration algorithm with GOP size 1, corresponding to processing each frame independently, and GOP size 16. Although this is the last frame of the group with the GOP size 16, the output of our algorithm with GOP size 16 in Figure 9(c), as compared with Figure 9(b) in which every frame is processed independently effectively, gains 4x speedups without visible

information loss.

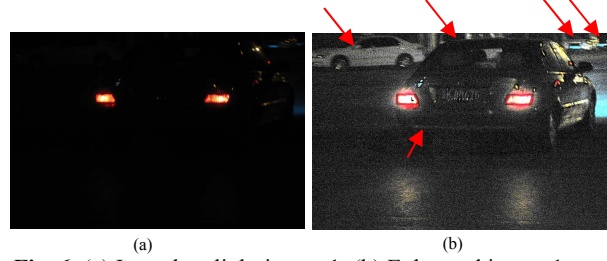


Fig. 6. (a) Input low light image 1. (b) Enhanced image 1 using proposed low lighting enhancement algorithm.



Fig. 7. (a) Input low light image 2. (b) Enhanced image 2 using proposed low lighting enhancement algorithm.



Fig. 8. (a) Input low light image 3. (b) Enhanced image 3 using proposed low lighting enhancement algorithm.

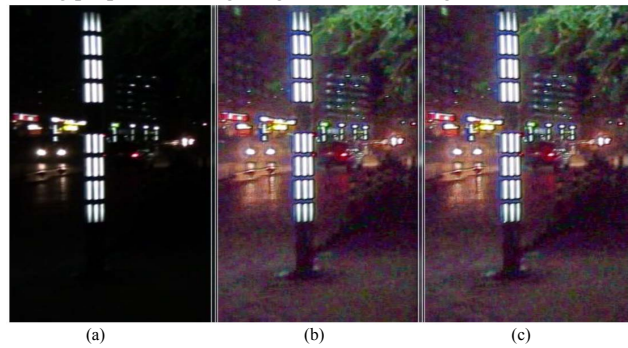


Fig. 9. (a) The 16th frame of the input video sequence. (b) Enhanced frame of (a) with GOP size 1. (c) Enhanced frame of (a) with GOP size 16.

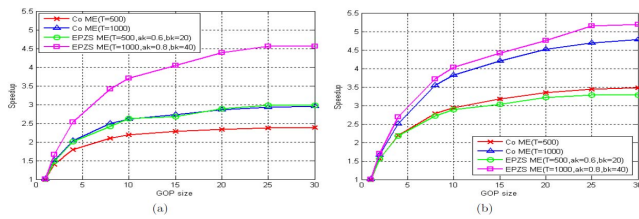


Fig. 10. Speedup of proposed fast enhancement algorithm using (a) Co ME and EPZS ME. (b) Co ME and EPZS ME with fast SAD algorithm.

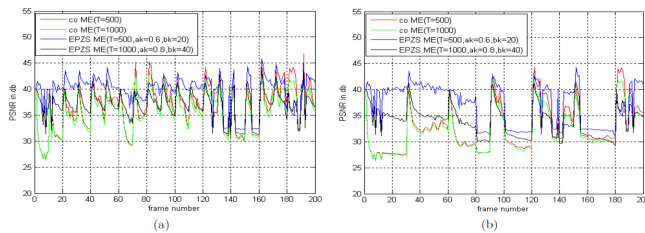


Fig. 11. PSNR of proposed low lighting enhancement algorithm using Co ME and EPZS ME. (a) GOP size is 10. (b) GOP size is 30.

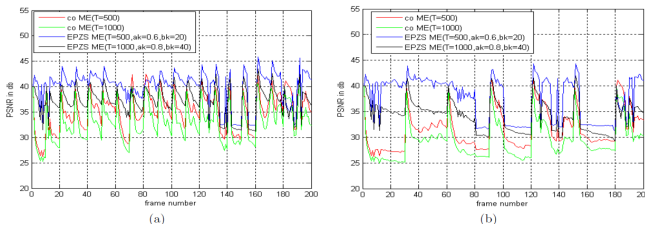


Fig. 12. PSNR of proposed low lighting enhancement algorithm using Co ME and EPZS ME with fast SAD algorithm. (a) GOP size is 10. (b) GOP size is 30.

5. CONCLUSIONS AND FUTURE WORK

In the paper, we described a novel, fast and efficient video enhancement algorithm for low lighting video. Based on the observation that inverted low lighting video are similar with haze videos, we change the de-hazing algorithm properly and apply it on the inverted video before inverting the de-hazed video again to produce the output enhanced low-lighting video. 4x speedups is achieved with no visible loss of critical information by performing ME and skipping the calculation of $t(x)$ for P frames.

Areas of further improvement include improvements in the de-haze algorithm and enhancement of videos in other weather conditions.

REFERENCES

[1] H. Ngo, L. Tao, M. Zhang, A. Livingston, and V. Asari. “A Visibility Improvement System for Low Vision

Drivers by Nonlinear Enhancement of Fused Visible and Infrared Video,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition.*, San Diego, CA, Jun. 2005, pp.25.

[2] “Night Driver, Making Driving Safer at Night” *Raytheon Company*, available at: <http://www.nightdriversystems.com/nightdriver.html>

[3] K. He, J. Sun, and X. Tang. “Single Image Haze Removal Using Dark Channel Prior,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition.*, Miami, FL, Jun. 2009, pp. 1956-1963.

[4] R. Fattal. “Single Image Dehazing,” in *ACM SIGGRAPH '08*, Los Angeles, CA, Aug. 2008, pp. 1-9.

[5] S. G. Narasimhan, and S. K. Nayar. “Chromatic Framework for Vision in Bad Weather,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition.*, Hilton Head, SC, Jun. 2000, vol. 1, pp. 1598-1605.

[6] S. G. Narasimhan and S. K. Nayar. “Vision and the Atmosphere,” in *Int. Journal Computer Vision*, vol. 48, no. 3, pp. 233–254, 2002.

[7] R. Tan. “Visibility in Bad Weather from A Single Image,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition.*, Anchorage, Alaska, Jun. 2008, pp. 1-8

[8] A. Zaccarin and B. Liu. “Fast Algorithms for Block Motion Estimation”, in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, San Francisco, CA, 1992, pp. 449-452.

[9] Y. L. Chan and W. C. Siu. “New Adaptive Pixel Decimation for Block Motion Vector Estimation,” in *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 113-168, 1996.

[10] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro. “Motion Compensated Interframe Coding for Video Conferencing,” in *Proc. Nut. Telecommun. Conf.*, New Orleans, LA, Nov. 1981, pp. G5.3.1-G5.3.5.

[11] B. Girod, and K. W. Stuhlmüller. “A Content-Dependent Fast DCT for Low Bit-Rate Video Coding,” in *Proc. IEEE Int. Conf. Image Process.*, Chicago, Illinois, Oct. 1998, vol. 3, pp. 80-83.

[12] A. M. Tourapis. “Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation,” in *Proc. Visual Communications and Image Processing.*, San Jose, CA, Jan. 2002, pp. 1069-1079.